Three-dimensional unstructured mesh generation: 
Part 1. Fundamental aspects of triangulation and point creation

Yao Zheng a, Roland W. Lewis a,*, David T. Gethin b

a Department of Civil Engineering, University of Wales Swansea, Swansea, SA2 8PP, UK
b Department of Mechanical Engineering, University of Wales Swansea, Swansea, SA2 8PP, UK

Received 9 February 1995; revised 9 June 1995

Abstract

The present paper introduces an alternative approach for Delaunay triangulation, in which the triangulation is mapped from an equivalent convex hull in a higher dimension. Furthermore, some fundamental aspects of a point creation algorithm for unstructured mesh generation have been addressed based on numerical experiments. A point spacing tensor and point insertion criterion have been introduced aiming to deal with anisotropic meshes.

1. Introduction

Mesh generation is a bottleneck problem in all finite element based numerical simulations, and has been continuously investigated for several decades since the emergence of the finite element method [1–3]. Various meshing methods emerged and some of them have become very popular and carry great promise. There are no algorithms absolutely superior to all others, and all the meshing methods seem to have their own advantages and drawbacks. Current literature abounds with references to various meshing methods and the applications of these meshes. Several major review works have appeared in the past along with the development of mesh generation methods [1–11].

One of the structured meshing methods is based on partial differential equations [12–15], which include elliptic, parabolic and hyperbolic systems. The motivation for the use of partial differential equations as mesh generators can be derived from a number of sources. Mainly, this type of approach utilizes some properties of the corresponding PDEs. For example, the nature of elliptic equations is to smooth boundary data, and this affords a most desirable property.

Algebraic mesh generation distinguishes itself from other mesh generation methods by its ability to provide a direct function description of the coordinate transformations between the computational and physical domains. This method includes transfinite interpolation [12, 14, 15], isoparametric mapping [15–17] and conformal mapping [15, 18, 19]. There are many formulations for both transfinite interpolation and isoparametric mapping in 2 and 3 dimensions, however, conformal mapping is normally considered in two-dimensional cases.

*Corresponding author.
Variational methods have been developed to minimize a global functional which takes different mesh properties into account. The general form of these variational methods allows a competitive enhancement of mesh smoothness, orthogonality and point concentration by representing each of these desired properties by integral measures over all the elements and minimizing a weighted average of them [15, 20–22]. The scope of this method used for optimization of structured meshes is also applicable to that of unstructured meshes [23].

In the structured meshing approaches mentioned above, the establishment of mapping between physical and regular transformed space has been the main idea. This implies that the computational domain is equivalent to a cube in topological sense. However, for general complex domains, there is an unavoidable restriction that an inappropriate mesh will be produced. Therefore, a multiblock subdivision of the domain needs to be introduced. The main idea behind this approach is to ensure that the connection between blocks is proper in the sense of valid connectivity [14, 16, 17, 24].

Of various unstructured meshing methods, spatial decomposition methods based on octree structures were originally proposed for use as approximate representations of geometric objects. The application of octree technique for three-dimensional mesh generation is to use the property of tree structures [25–29]. This method provides the advantage that the hierarchically structured octants are easily created and accessed, and the useful data structure facilitates the implementation of the mesh generator. During the recursive subdivision of a bounding parallelepiped, it is possible to create octants containing a disproportionately small piece of the object which can lead to poorly shaped elements in the final mesh if no quality improvement procedure is undertaken. Some mesh generators use the octant structures and Delaunay triangulation, while the others employ the octree structures alone.

Much of the initial development of Delaunay mesh generation algorithms have been built on the Delaunay property [30–32] and the associated computational methods [30–34]. With these properties, various algorithms have been established to triangulate a set of points in the dimensions concerned. Delaunay type mesh generators have become increasingly popular [14, 26, 29, 35–38].

Delaunay triangulation only provides a scheme to connect the points to form elements, however algorithms have to be established to create the points. Among these approaches of generating points, the octree method has proven to be very useful [26, 29]. Subsequently, Steiner point creation scheme can also be used to generate points for triangulation. The resulting triangulation is referred to as Steiner triangulation, which can be obtained via Delaunay triangulation [2, 38]. The quality of resulting meshes is quite promising.

Over the past several years, element-by-element removal procedures called as advancing front [14, 39–43] and paving (plastering) [44, 45] techniques have received considerable attention. In these methods the meshing procedure begins at the boundary and develops elements working into the interior of the domain until it is filled. Two reasons for the popularity of these approaches are the ability to reflect directional mesh gradation information and the fact that the resulting meshes are sensitive to the boundary of the domain. Using this method, the surface elements are usually well formed, though some poorly-shaped elements may be produced away from boundaries where the fronts merge.

There have been a number of approaches to automatic mesh generation developed, based on partitioning the domain of interest into sets of subdomains and subsequently meshing those subdomains. A recent efficient method, referred to as the medial [46–48] or symmetric [49] axis transformation, employs the basic concept of a Voronoi diagram to define subdomains. This approach has the advantage of reducing the domain of interest into a number of fairly simple subdomains step by step.

Recently, there have been some applications of artificial intelligence techniques for mesh generation which appear to assist the user to some extent. Some rules for deciding control densities for finite element meshes can be inductively constructed from examples provided by experts [50]. The expert system generated in [51] was able to intelligently identify critical regions and choose a proper mesh size. Using a neural network technique, a system was developed to predetermine the mesh density. This system can be trained by incorporating examples of ideal meshes [52, 53].

In the present paper, an alternative approach to Delaunay triangulation is to be addressed, in which the triangulation is obtained from a corresponding convex hull in a higher dimension. Within this type of triangulation, point creation algorithms play an important role in terms of quality of resulting meshes and performance of the mesh generator. It is intended to study a point creation algorithm by means of
numerical experiments. Because the interior points are created based upon the previous triangulation, the point creating algorithm is to be presented after Delaunay Triangulation. This paper forms a foundation for surface and volume meshing methods to be mentioned in other papers in this series [54, 55].

2. Delaunay triangulation

2.1. Voronoi diagram

Dirichlet in 1850, first proposed a method in which a domain could be systematically decomposed into a cell complex, i.e. a set of convex polytopes [56]. Let S be a finite set of sites (i.e. points) of Euclidean space $E^d$, and for each $s \in S$ let $d_s$ be a mapping of $E^d$ to the positive real numbers; the term $d_s(p)$ (this may also be written as $d(p,s)$) is the Euclidean distance between a site $s$ and a point $p$. The set $\{ p \in E^d | d_t(p) < d_s(p), t \in S-\{s\} \}$ is the Voronoi cell of $s$, and the cell complex, defined by the Voronoi cells of all sites in $S$, is called the Voronoi diagram $\mathcal{V}(S)$ of $S$ [31, 57, 58]. There is abundant mathematical literature on Voronoi diagrams and several independent investigations have lead to alternate names, e.g. Dirichlet tessellations, Thiessen polygons, and Blim’s medial axis transform [31]. This mathematics concept and its associated algorithms have been studied for more than one century. Okabe et al. [59] addressed several application areas such as spatial interpolation, models of spatial process, point pattern analysis and locational optimization.

2.2. Delaunay triangulation

The polytopes, obtained by connecting a straight-line segment between each pair of sites of $S$, whose Voronoi polytopes share an edge, is referred to as a straight-line dual of the Voronoi diagram. Delaunay proved that this dual is a triangulation of $S$ [60]. In this simple form the theorem fails if certain subsets of four (2D case) or five (3D case) or more points are cocircular (cospherical). However, in this case the completion of the triangulation is relatively straightforward [30, 32]. The corresponding triangulation is referred to as Delaunay triangulation $\mathcal{DT}(S)$. Voronoi diagram and Delaunay triangulation are seen as two sides of the same coin, many applications utilize this property in their algorithms.

Let $S$ be a finite set of points in the plane. It can then be proved that a triangulation $T$ of $S$ is globally equiangular if, and only if, if it is a completion of Delaunay triangulation $\mathcal{DT}(S)$ [31]. In higher dimensions, there is a similar property, which forms the base for mesh generation using Delaunay type triangulation.

Each element of the resulting mesh satisfies the Delaunay criterion, i.e. the circumcircle (circumsphere in three dimension) of the element contains no other points. This type of triangulation of a point set results in a convex hull, which covers all these points. There are several algorithms in the literature [33, 34, 61, 62], which demonstrate the applicability of Delaunay triangulation.

2.3. Delaunay triangulation and convex hull

The relationship between the Voronoi diagram in $E^d$ and a particular convex polytope in $E^{d-1}$ was first exploited by Brown [63] for designing algorithms that work in arbitrary dimensions. The transformation, which maps the problem of constructing Delaunay triangulation into that of constructing convex hulls, is well known [31, 63, 64]. A Delaunay triangulation can be obtained from an equivalent convex hull in a higher dimension by means of a geometric transformation.

Let a generic point in real number space $R^d$ be represented by $p_i$, then its $d$ coordinates can be represented by $\{p_{ij}, j = 1, \ldots, d\}$. Let $S$ describe the set of $(d + 1)$ test points $\{p_i, i = 1, \ldots, d + 1\}$ defining a simplicial element (polytope) in $R^d$, and let $p_l$ be an additional point to be tested for inclusion in the $d$-dimensional sphere determined by the points in $S$. Correspondingly, let $S'$ define a set of $(d + 1)$ points $\{p_{il}, i = 1, \ldots, d + 1\}$ in $R^{d+1}$ obtained by subjecting each of the points in $S$ to the following coordinate transform $\mathcal{PR}$:
\[
\begin{aligned}
    p'_{ij} &= p_{ij}, \quad j = 1, \ldots, d; \\
    p'_{ij} &= \sum_{k=1}^{d} p_{ik}^2, \quad j = d + 1,
\end{aligned}
\]

(1)

where \( p'_i \) is represented as \( \{p'_{ij}, j = 1, \ldots, d + 1\} \). Likewise, map \( p_i \) into \( p'_i \) in \( \mathbb{R}^{d+1} \). This mapping simply lifts each point vertically upwards until it lies on the paraboloid of revolution as defined by equation

\[
\sum_{k=1}^{d} p_{ik}^2 = p_{d+1}
\]

(2)

and it is shown in Fig. 1.

The equivalence between the two problems can be stated as follows. The condition that the point \( p_i \) lies in the hypersphere determined by the points in \( S \) is exactly the same as that the point \( p'_i \) is located below the hyperplane in \( \mathbb{R}^{d+1} \) determined by the points in \( S' \).

Therefore, to compute the Delaunay triangulation of a point set \( P \), it is sufficient to apply the mapping \( \mathcal{PR} \) to each of these points, construct the convex hull of the resulting point set \( P' \), and inversely map these points back to their corresponding origins by dropping the last coordinates. In essence, the Delaunay triangulation is the dual of the nearest-point Voronoi partition of the points, the elements of the mesh are the ‘downward’ faces of the convex hull; while the ‘upward’ faces of the convex hull are the dual of the furthest-point Voronoi partition [31].

It should be mentioned that there are other types of mapping capable of circumventing the problems between creating Delaunay triangulation and constructing convex hulls. Fig. 2(a) illustrates a parabolic mapping onto a mesh, while Fig. 2(b) shows a sphere mapping which is due to Brown [63]. The advantage of the parabolic transformation over that of the unit-sphere is that the former is of capability to map sites with integer coordinates to hyperplanes with integer coefficients [31]. Fig. 3 illustrates an example of the mapping between an equivalent convex hull touching a paraboloid of revolution and its corresponding Delaunay triangulation, where 150 points are generated randomly within a square and 4 points are the corner of the square. In this convex hull, one more point above all the projected points is introduced, which represents the point at infinity.

Fig. 1. The 3-dimensional version of the parabolic mapping.
Fig. 2. A graphical illustration of the geometric transformations on a mesh: (a) parabolic mapping; (b) spherical mapping.

Fig. 3. A 3-D convex hull with its corresponding 2-D Delaunay mesh.
2.4. Creating convex hulls

The advantage of transforming the Delaunay triangulation problem into the convex hull problem is that more results are available for the latter in the area of computational geometry. For the general case of computing convex hulls in an Euclidean space of more than two dimensions, there exist five basic algorithms, based on different concepts, which are: the Gift-Wrapping method, the Beneath-Beyond method, the Divide-and-Conquer method, the Shelling method and the Quickhull method, respectively [30–32, 64, 65].

The ‘Gift-Wrapping’ method is the oldest method due to Chand and Kapur [66]. The idea is to take an infinitely stretchable, weightless piece of \((d−1)\)-dimensional wrapping foil and wrap it around the point set under consideration. This method uses a \((d−2)\)-facet \(f\) of an already constructed \((d−1)\)-facet \(F_1\) to create the adjacent \((d−1)\) facet \(F_2\), which shares \(f\) with \(F_1\). The complexity of this method is claimed as \(O(n^{\lfloor\frac{d}{2}\rfloor+1}) + O(n^{\lfloor\frac{d}{2}\rfloor} \log n)\) [30, 64].

A second technique, called the ‘Beneath-Beyond’ method, has been invented by Kallay [30, 64]. This algorithm incrementally constructs the convex hull by adding a point at one time. Consider a point \(p\), if it is external to the current convex hull \(\text{conv}P\), then construct the supporting ‘cone’ of \(\text{conv}P\) from \(p\) and remove the portion which falls within the ‘shadow’ of this cone. The corresponding complexity is \(O(n^{\lfloor\frac{d}{2}\rfloor}) + O(n \log n)\) [31, 64].

The ‘Divide-and-Conquer’ method divides the problem recursively into two subproblems of nearly equal size, i.e. two convex hulls are constructed separately but recursively and are then merged. The corresponding complexity is reported as \(O(n^{\lfloor\frac{d}{2}\rfloor}) + O(n \log n)\) [31, 64].

The ‘Shelling’ method is due to Seidel [67]. The central idea is to use the concept of a shelling line and the complexity is \(O(n^{\lfloor\frac{d}{2}\rfloor} \log n)\) [64].

The main idea behind the ‘Quickhull’ method is to introduce a Quicksort mechanism in partitioning the point set. The original techniques are due to the strategies proposed independently by Eddy [68], Bykat [69], Green and Silverman [70] and Floyd [30]. These techniques have been generalized by combining with the general dimensional Beneath-Beyond method [65]. The expected performance of the general Quickhull is an open problem yet, however, it is claimed that the algorithm is simple, uses less memory and allows good use of virtual memory [65].

2.5. Constructing Delaunay triangulations

The Quickhull algorithm has been used herein, and the following terminology describing the method is outlined [65]. For a \(d\)-dimensional problem, a \((d−1)\)-dimensional facet is endowed with its normal. If a point is located on the same side as the normal direction with respect to the facet, then the point is referred to as being situated above the facet, otherwise it would be below the facet. A visible facet for a point is one that is below the point. A horizon for a point is the boundary of its visible facets and consists of horizon ridges. A new facet related to a point is a facet with the point as its apex and a horizon ridge as its base, and a cone for a point is the set of new facets.

Fig. 4 shows the incremental procedure to create a 3-D convex hull. Fig. 4(a) is a convex hull of 50 points randomly located on a sphere of radius 0.5. A further point (0.5, 0.5, 0.5) is introduced, which can be seen at the upper right corner of each frame of Fig. 4(b)–(f). Fig. 4(b) and (c) show the visible and invisible facets, respectively, from the point, whereas (d) depicts the horizon facets which border the visible facets. The inside edges of the horizon facets are horizon ridges. In Fig. 4(e) a cone is formed by linking the point with the horizon ridges. Finally, the new convex hull is constructed by combining the cone with the invisible facets as shown in Fig. 4(f).

In the Quickhull method, when a cone of new facets is created, the visible facets for each point are subsequently updated. This process is termed partitioning as it is similar to the partition step of Quicksort. The outside set for a facet contains the points which are above the facet and the facet is visible from each of these points. Partitioning re-assigns the outside sets of replaced facets and those of
their immediate neighbours. It rejects points of replaced facets and unprocessed points if the point is inside the cone. Partitioning also records the furthest point of each outside set.

A point is called an 'extreme point' of a point set if it is a vertex of the corresponding convex hull. In the equivalent convex hull of a Delaunay triangulation in a lower dimension, all points are extreme points. This is the feature of the equivalent convex hull, which leads to a special treatment. Therefore, the equivalent algorithm has smaller complexity than a general convex hull procedure does.

For three-dimensional mesh generation, the geometry can be specified by means of the surface patches. Firstly, points are to be generated along the edges of these patches according to the point spacing requirement. The second step is to generate a surface mesh from the points on the edges and new points located inside the patches. The triangulation of every surface patch is carried out on a parametric plane which is related to the surface patch in the geometrical transformation sense. The points situated inside the patches are generated during the construction of the equivalent 3-D convex hull, taking account of the point spacing parameters. The third step is to generate a three-dimensional mesh from the nodes of the surface mesh and the 8 extra points at the corners of a bounding box. All the interior points within the volume are generated according to the global point spacing requirement while the equivalent 4-D convex hull is being constructed. After that, the fifth step is to carry out the boundary recovery for the whole volume or multi-material domains.

The third step is an analogy to the second, in which a part of the points are generated prior to the triangulation process whereas the rest are created during the triangulation. The algorithm for general d-D triangulation can be listed as follows.
Algorithm 1

Input initial point set \( P_1 \) in \( d \)-D real space
Create the corresponding bounding polytope
Generate an extra point set \( P_b \) from the bounding polytope
Map \( P_1 \cup P_b \) onto \( P'_1 \cup P'_b \) on a paraboloid of revolution
Create an initial hull in \((d + 1)\)-D space for \((d + 1)\) points of \( P'_1 \cup P'_b \)
Partition the remaining points into the outside sets of the initial hull
For each facet with a non-empty outside set (Loop 1)
  Select the furthest point of the set
  Find the horizon and other visible facets for the point
  Make a cone of new facets from the point to the horizon
  Partition the outside sets with respect to the cone
For each facet of the current convex hull (Loop 2)
  Accept the centroid of the facet if it is a required node
  (This is conducted by taking account of point spacing parameters.)
  Group these accepted points into a new point set
For each facet with a non-empty outside set (Loop 3)
  Find the horizon for the point
  Construct a cone of new facets from the point to its horizon
  Replace the facets inside the cone
  Update the convex hull
Repeat Loops 2 and 3
  (All the centroids of the facets accepted forms \( P'_1 \))
Obtain a \( d \)-D Delaunay triangulation from the \((d + 1)\)-D convex hull
  (The resulting triangulation is supported by \( P_1 \cup P_2 \cup P_b \))

In the above algorithm, Loop 1 is designed to create a convex hull for \( P'_1 \cup P'_b \) incrementally. Loop 2 is the point creation part, of which the details are to be given in Section 3. And Loop 3 is to process the accepted points created by Loop 2. It can be seen that the interior points other than that in \( P_1 \) are generated batch by batch, each of the batches is followed by a triangulation procedure (Loop 3). Every time to enter Loop 2 for creating a batch of points is referred to as a point creation level.

3. Point creation

3.1. Introduction

In various meshing algorithms, points can be generated in many ways [1–3], which can be classified into two main types: dependent and independent techniques. The former has been used in Delaunay triangulation [3, 38], advancing front [39, 41–43], and so on, where the points are created with the aid of connectivity information. The latter is an approach in which points are generated independently of connectivities, as used in some quadtree and octree methods [6, 29], and structured meshing techniques [12].

In the present work, a dependent method has been adopted following that introduced in [38]. Also, a scheme [29] using octree domain decomposition to generate points independently before triangulation has been investigated. From Fig. 5, it is observed that the current scheme gives better quality meshes than the octree domain decomposition method can do if no smoothing procedure is performed. The octree method can result in nice adaptive meshes as mentioned in the literature (e.g. [3, 29]). Particularly, when mesh size desired changes greatly, the mesh size of the resultant appears with smooth change as required. However, the current approach gives greater possibility to create elements with smoothly changing sizes when compared with the octree method.
3.2. The algorithm

When the element size required in some regions are given, the point spacing function can be derived by means of interpolation. Several interpolation formulations can be used in this algorithm. The value at a particular point is to be influenced by the spacing control points in its local region or in the whole domain. In order to control the point spacing locally, the radius of a search kernel $s$ can be defined (Fig. 6). For a point of interest $P$, if there are spacing control points in the search kernel of radius $s$, then the interpolation will be conducted upon these spacing control points, otherwise $s$ is increased by doubling it.

Assuming points on the edges of a surface patch are created, an interior point of the patch is to be generated, being controlled by the point spacing function values of vertices of an intermediate surface element, and this point is positioned at the centroid of the element. Then the point is endowed with a value of point spacing. Similarly, an interior point of a volume mesh can be generated by taking account of the point spacing function values of vertices of an intermediate volume element. The algorithm is given as follows for both surface and volume meshes. In the following list, $m = 1, \ldots, 4$ is for the case of creating interior points of volume meshes. To generate interior points of surface meshes, $m = 1, 2, 3$ is to be used.

Fig. 6. Search kernels of interpolation of point spacing function.
ALGORITHM 2

Give boundary points associated with their point spacing function values
Triangulate these boundary points
Initialize the number of interior points created \( N = 0 \)
For each element within the domain concerned (Step A)
(An element: a facet of the corresponding convex hull in a higher dimension)
Assign a prospective point \( Q \) to be the centroid of the element
Calculate the value of point spacing function \( d_Q \)
\begin{itemize}
  \item by interpolating from the values of vertices of the element
  \item Derive the distance \( r_{dm}(m = 1, \ldots , 4) \) from \( Q \) to the vertices
  \item If \( (r_{dm}/d_Q < \alpha \lambda) \) for one of \( m = 1, \ldots , 4 \), then
    \begin{itemize}
      \item reject point \( Q \); do the next element
    \end{itemize}
  \item Compute distance \( r'_{dj} \) from \( Q \) to other points \( P_j(j = 1, \ldots , N) \)
    \begin{itemize}
      \item If \( (r'_{dj}/d_m < \beta \mu) \) for one of \( j = 1, \ldots , N \) and one of \( m = 1, \ldots , 4 \), then
        \begin{itemize}
          \item reject point \( Q \); do the next element
        \end{itemize}
    \end{itemize}
  \item Accept point \( Q \) as an interior point and include \( Q \) in the list \( P_j(j = 1, \ldots , N) \)
  \item Assign point spacing function value \( d_Q \) to \( d_N \) for new point \( P_N \)
\end{itemize}
If \( N = 0 \), then exit from this point creation procedure
Otherwise, perform triangulation and go to Step A

In the above algorithm, coefficients \( \alpha \) and \( \beta \) are control parameters for the user to set. \( \alpha \), with default value 1.0, controls the point spacing by changing the allowable shape of formed element, whereas \( \beta \) is a mesh regularity parameter. The effects of regularity parameter \( \beta \) is demonstrated in Fig. 7. The first meshes in (a) and (b) are identical, and are triangulation based on edge points. The second and third meshes in both series correspond to meshes at point creation levels 1 and 3. The last ones in both cases are final meshes. Based on these experiments, the default value of \( \beta \) is set to be 1.0. Coefficients \( \lambda \) and \( \mu \) are constant factors in the algorithm, their values are to be deduced in Section 3.5. As a result, we have \( \lambda = \mu \approx 0.70228615 \) for creating points on surface patches, and \( \lambda \approx 0.91925265, \mu \approx 0.6128351 \) for creating points in a volume.

The main idea behind Algorithm 2 is to generate points with regard to the point spacing and element shapes. Criterion \( r_{dm}/d_Q < \alpha \lambda \) is designed to control the point spacing with reference to the vertices of

![Fig. 7. The effect of regularity parameter: (a) \( \beta = 10.0 \); (b) \( \beta = 0.1 \).](image)
the parent element of point $Q$. Criterion $r_{ij}'/d_m < \beta \mu$ is related to the element shapes, and it avoids the case that point $P_j$ is inside a certain circle centered at point $Q$. In the triangulation, the location of point $P_j$ effects the element construction directly due to the Delaunay property being present.

In Algorithm 2, for a point, its point spacing function is based on interpolation from the vertices of an intermediate element. However, other interpolation formulations can be employed to reflect local geometry features. Weatherill and Hassan [38] introduced line and point sources for this purpose. Furthermore, the interpolation scheme can also be modified to incorporate with adaptivity.

Delaunay triangulation assumes the points are located in 'general positions', that is, any five of them are not cospherical for three-dimensional cases. Although some treatment can be made to remove the difficulty of this degeneracy, general positioning is still preferable. From this point of view, this point creation scheme is more handleable comparing with octree domain decomposition, since the points created using this scheme are located at the centroids of intermediate elements (triangles for 2-D or tetrahedra for 3-D).

Numerical experiments demonstrate good quality of resulting meshes using this point creation scheme, although its mathematical proof is an open problem yet to be solved. There is an alternative method which may be employed to create the interior points generated during the Delaunay triangulation. In this method the points are located in the centres of the circumspheres of intermediate elements. Dey et al. [71] gives a mathematical proof that a mesh of good quality is to be obtained by using this method. However, there are limitations found in the current work, which are to be mentioned as follows. In order to make point insertion compatible with the point spacing value required, the perspective point should be checked against the distances to the nearby vertices. As a circumcenter is sometimes outside the associated intermediate element, it is more expensive to search near vertices than that using a centroid. Additionally, some of the circumcenters even are outside the bounding box from which the triangulation started. If these circumcenters are accepted, then possibly more temporary elements will be introduced. If these circumcenters are not be considered, then unfavourable point spacing will ensue at some point creation stages. Therefore, special care should be taken to ensure the feasibility of this scheme.

3.3. Note on history

This point creation scheme together with a triangulation used can be referred to as Steiner triangulation, because there is a mathematics problem named after Steiner, which addressed this point insertion scheme [59]. According to Kuhn [72], this problem dates back to Fermat in the 17th century, and was introduced in [73] as the Steiner (Steiner's) problem.

To give an example of the Steiner problem, suppose that there are three sites on a plane where we can freely lay out roads for them (Fig. 8(a)). The problem is to construct a network of roads to connect these sites with the minimum length of roads, provided that the roads are directly connected between these sites. One solution can be given by an Euclidean minimum spanning tree as shown in Fig. 8(b). However, this solution is not optimal if a new site is allowed to be added. To determine optimal location of the new site, the Euclidean Steiner minimum tree provides a solution (Fig. 8(c)), in which the new site is referred to as a Steiner point. For this particular case, the Steiner point is situated at the centroid of the triangle formed from the given sites.

Fig. 8. An example of the Euclidean minimum spanning tree (b) and the Euclidean Steiner minimum tree (c).
3.4. Creating points on patch edges

The point creation algorithm mentioned above is related to the points on surface patches or those in a volume, rather than the points on patch edges. The generation of points on patch edges is performed separately, and it is the starting point of the mesh generation. These points can be introduced by subdividing the edge from one end to the other, where the point spacing function is a factor to control the intervals. A general way is to use arc length $dl$ to check against a value of point spacing function (refer to Fig. 9(a)). A subdivision can be obtained such that the corresponding $dl$ is as expected, if enough iterations are allowed. For a curvilinear edge, parameter coordinate is helpful in this procedure.

For an edge of high curvature, a great computing cost is required. In order to reduce this cost, a binary tree has been adopted to subdivide the edge in a similar manner to the quadtree in a plane and octree in a volume (Fig. 9(b)). Every subdivision is performed by introducing a central point (in a parametric coordinate) of an edge cell. The resulting cell sizes on an edge change step by step, a less smooth change in arc length is obtained when compared with that from the general approach. However, the numerical experiments in the next subsection demonstrate its effectiveness with an advantage of lower computing cost.

With the binary tree subdivision, whether a cell should be subdivided is subject to a check on cell size against the required spacing function. Here, cell size is taken as the straight line distance between the ends of the cell rather than arc length for the sake of simplicity. The corresponding numerical experiments incorporated with this treatment show reasonable results (Fig. 13).

Consider a straight edge of length $L$ and desired cell size $l_d$. The number of cells takes one value from 1, 2, 4, 8, 16, .... As Fig. 10 shows, there are three criteria of subdivision, where (a) and (b) are extreme cases while (c) is supposed to be an optimal case. Let $n_0$ be the number of cells, then the following hold:

$$\frac{L}{2n_0} < l_d < \frac{L}{n_0}, \quad \text{for scheme (a)}; \quad (3)$$

$$\frac{L}{2n_0} < \frac{l_d}{2} < \frac{L}{n_0}, \quad \text{for scheme (b)}; \quad (4)$$

$$\frac{L}{2n_0} < fl_d < \frac{L}{n_0}, \quad \text{for scheme (c)}, \quad (5)$$

where $f$ is a coefficient to be determined. Regarding the area of shading in the figure for these schemes, the corresponding areas are $L$, $L/2$ and $fL$, respectively. On the other hand, the area of the bounded region corresponding to the optimal scheme should be

![Fig. 9. Two schemes of point creation on an edge.](image-url)
\[ A_{\text{opt}} = \int_{n_0}^{2n_0} \frac{L}{x} \, dx = L \ln x \bigg|_{n_0}^{2n_0} = L \ln 2. \]  

As Scheme (c) is the optimal one, there exists \( fL = L \ln 2 \) or \( f = \ln 2 \). This means that when the ratio of cell size to desired one \((L/n_0/L_d)\) is less than \( 2f \approx 1.3862944 \), the subdivision process completes.

3.5. Deduction of constant factors

Points on surface patches

In the point creation routine for surface patches, the constant factors \( \lambda \) and \( \mu \) are involved, which can be derived based on a case where equilateral elements are presented as shown in Fig. 11. Before point \( P \) is introduced, the average area of these four elements is \( A \). After point \( P \) is inserted, the average area (of
Fig. 12. Spacing of newly introduced point to its neighbours.

Fig. 13. Element sizes of surface meshes in a square with edge length 2.0.
six elements) is $\frac{2}{3}A$. Referring to Fig. 11, assume the function $y = (\frac{2}{3})^x A$ is a reasonable approximation of the average area in a small region consisting of the element and its neighbours. In an analogy to the deduction in Section 3.4,

$$H = \int_0^1 \left(\frac{2}{3}\right)^x A \, dx \approx 0.82210115A$$

such that the function $y = H$ bounds the same shading area as $y = (\frac{2}{3})^x A$ in the interval $x \in [0, 1]$.

The ratio $A/H$ has been introduced as a contribution to $\lambda$ in Algorithm 2, and $\lambda = \lambda_1 \lambda_2$, where $\lambda_1 = A/H \approx 1.2163953$, $\lambda_2 = L_1/L = \sqrt{3}/3$. With reference to Fig. 12, the coefficient $\mu$ has been determined as $\mu = \mu_1 \mu_2 \mu_3$, where $\mu_1 = \lambda_1$, $\mu_2 = \lambda_2$ and $\mu_3 = 1$.

Fig. 13 shows the compatibility between points generated on edges using the binary tree method and points on a surface patch. This confirms the reasonableness of the $\lambda$ and $\mu$ values chosen. In this plot, log scaling on both axes has been applied.

**Points in a volume**

In order to study the factors $\lambda$ and $\mu$ in Algorithm 2 for volume meshes, five equilateral tetrahedra with the same volume $V$ are considered. Referring to Fig. 14, the average volume of the 12 elements is $\frac{5}{12}V$. By analogy, the coefficient $\lambda$ is chosen as $\lambda_1 \lambda_2$, where

$$\lambda_1 = 1/\int_0^1 \left(\frac{5}{12}\right)^x \, dx \approx 1.5008035, \quad \lambda_2 = \frac{L_1}{L} = 0.612507.$$  \hspace{1cm} (8)

Furthermore, the coefficient $\mu$ is taken as $\mu_1 \mu_2 \mu_3$, in which $\mu_1 = \lambda_1$, $\mu_2 = \lambda_2$. Since the distance between two centroids are $\frac{2}{3}L_1$, $\mu_3$ is chosen to take the value $\frac{2}{3}$.

Table 1 shows the mesh characteristic parameters, which indicates that the choice of $\lambda$ and $\mu$ in Algorithm 2 is reasonable. In this table, the desired element size means desired side length, the optimal area and volume are corresponding to equilateral elements.

### 3.6. Point spacing tensor and point insertion criterion

In certain application areas, such as shocks and boundary layers, an anisotropic mesh is expected to be employed [74, 75]. A brief study is to be taken in this direction of research, in which two new concepts, point spacing tensor and point insertion criterion, are introduced.

In Algorithm 2, the point spacing information is the base of a judgement as to whether a prospective point should be inserted or not. The criterion for this can be restated as follows:

---

Fig. 14. Introducing a new point to an equilateral tetrahedron.
Table 1
Mesh characteristic parameters for a cube with edge length 2.0

<table>
<thead>
<tr>
<th>Desired element size</th>
<th>Edge cell size</th>
<th>Surface elements Average area</th>
<th>Optimal area</th>
<th>Volume elements Average volume</th>
<th>Optimal volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>1.05e-1</td>
<td>1.21e-1</td>
<td>1.08e-1</td>
<td>1.92e-2</td>
</tr>
<tr>
<td>0.25</td>
<td>0.25</td>
<td>2.75e-2</td>
<td>3.00e-2</td>
<td>2.71e-2</td>
<td>1.95e-3</td>
</tr>
<tr>
<td>0.125</td>
<td>0.125</td>
<td>6.58e-3</td>
<td>7.38e-3</td>
<td>6.72e-3</td>
<td>2.35e-4</td>
</tr>
<tr>
<td>0.0625</td>
<td>0.0625</td>
<td>1.75e-3</td>
<td>1.88e-3</td>
<td>1.69e-3</td>
<td>—</td>
</tr>
<tr>
<td>0.03125</td>
<td>0.03125</td>
<td>4.11e-4</td>
<td>4.20e-4</td>
<td>4.23e-4</td>
<td>—</td>
</tr>
</tbody>
</table>

If \( r_d/d < 1 \), then reject the prospective point, where \( r_d \) is a distance between this point and an associated point, and \( d \) is point spacing function value.

For an orthotropic mesh, the point spacing tensor is defined as

\[
D = D_{ij} \quad (i, j = 1, 2, 3)
\]

of which the principal components are \( (d_1, d_2 \) and \( d_3 \)), and their principal directions are \( (l_1, m_1, n_1) \), \( (l_2, m_2, n_2) \) and \( (l_3, m_3, n_3) \), respectively. The scalar value \( r_d \) can be generalized to a vector \( (r_{dx}, r_{dy}, r_{dz}) \), which has an alternative representation \( (r_{d1}, r_{d2}, r_{d3}) \) in the new coordinate system consisting of the principal directions. Therefore,

\[
\begin{bmatrix}
 r_{d1} \\
 r_{d2} \\
 r_{d3}
\end{bmatrix} =
\begin{bmatrix}
 l_1 & m_1 & n_1 \\
 l_2 & m_2 & n_2 \\
 l_3 & m_3 & n_3
\end{bmatrix}
\begin{bmatrix}
 r_{dx} \\
 r_{dy} \\
 r_{dz}
\end{bmatrix}.
\]

Furthermore, the point insertion criterion can be generalized to

\[
\left( \frac{r_{d1}}{d_1} \right)^2 + \left( \frac{r_{d2}}{d_2} \right)^2 + \left( \frac{r_{d3}}{d_3} \right)^2 < 1.
\]

In a finite element simulation, the point spacing tensor can be computed by means of coordinate transformation, provided the characteristic information of directional meshing is given.

Delaunay triangulation always results in meshes with the property that the meshes are optimal if the geometry boundary doesn’t restrict the triangulation. In 2D cases, the triangulation can maximize the minimal angles of the triangles. This means that anisotropic meshes are not obtainable if the Delaunay property is present.

Assume a domain is stretched first and then a mesh is generated by means of Delaunay triangulation. If a transformation is performed to stretched the domain back to its original shape, then the resulting mesh will be anisotropic. On the other hand, the distance between two points, used in this common Delaunay triangulation, is defined as Euclidean distance measure, which is referred to as uniform distance measure herein. An idea is raised straightforward that non-uniform distance can be employed in the Delaunay triangulation, in order to obtain an anisotropic mesh. This non-uniform distance measure comes from a generalization of the uniform distance measure as follows

\[
\text{dist} = k \sqrt{\left( \frac{r_{d1}}{d_1} \right)^2 + \left( \frac{r_{d2}}{d_2} \right)^2 + \left( \frac{r_{d3}}{d_3} \right)^2},
\]

where \( k \) is a coefficient globally defined, and \( d_i \) \((i = 1, 2, 3)\) is a representative value for this pair of points.

Fig. 15 illustrates meshes obtained by the Steiner triangulation (i.e. the current meshing program), given the orthotropic mesh requirement as shown in Fig. 16. The resulting meshes are isotropic in these two cases, in which the parameter \( f \) is chosen as 0.1 and 0.05, respectively, for the narrow rectangular area, whereas it is always given value 1.0 for the rest of the area. It has been observed that the transient between fine and coarse element areas are not smooth. The cause of this is that the point spacing parameter on a prospective point during the point creation is directly given as shown in Fig. 16, which presents a sharp change of the point spacing parameter in the domain concerned. If the point are created
in such a way that the point spacing parameter is obtained by an interpolation of the corresponding parameters on the vertices of the intermediate parent element, then the transient area of the resulting mesh will be smooth. The two meshes, as shown in Fig. 15, are isotropic due to the fact that Delaunay property prevents them from being anisotropic when uniform distance measure is adopted.
4. Conclusions

An alternative approach for Delaunay triangulation has been introduced in the present paper, in which the triangulation is mapped from an equivalent convex hull in a higher dimension. All the algorithms for constructing convex hulls can be employed to create triangulations. In the equivalent convex hull, all points are extreme points. This feature makes a special treatment possible so that the equivalent convex hull algorithm has smaller complexity than the general procedure.

Steiner points of intermediate elements can be introduced as interior points during Delaunay triangulation. In this sense, the resulting mesh can be referred to as Steiner triangulation. Considerations have been made to find suitable control parameters involved in the point creation algorithm. Numerical experiments demonstrate the effectiveness of the scheme.

Finally, a point spacing tensor and point insertion criterion have been introduced aiming to deal with anisotropic meshes. It has been discovered that, if the points are triangulated subject to the Delaunay property with assumption of uniform distance measure, the resulting meshes are not anisotropic in nature.

Acknowledgements

This work has been carried out under financial support by the Engineering and Physical Science Research Council of the United Kingdom (Grant Number GR/G 35107), which is gratefully acknowledged. The geometry visualization programs used in this work are GeoView [76] and FEView [77]. FEView, making use of Forms GUI library [78], works with GeoView for finite element applications. We would like to thank the creators of GeoView for permission to use their software, and Prof. Mark Overmars for the use of the Forms Library.

References


